COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
HW 2

Submit your codes and answers to Canvas for the problems given below.

1. Consider sorting n numbers stored in array A by first finding the smallest element of A and exchanging it with the element in $A[1]$. Then find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first n elements of A. This is called the ***selection sort*** algorithm.

(a) Write a pseudo-code for this algorithm

(b) What loop invariant does this algorithm maintain? State a loop invariant and prove that the algorithm works correctly.

(c) Give the best-case and worst-case running times of this algorithm as a function of n.

2. Observe that the **while** loop of lines 5–7 of the INSERTION-SORT procedure given below uses a linear search to scan (backward) through the sorted subarray $A[1..j-1]$. Can we use a binary search instead to improve the overall worst-case running time of insertion sort to $\Theta(n \lg n)$? What if a doubly linked list is used instead of an array? Explain the reason.

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1]
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

3. We can express insertion sort as a recursive procedure as follows. In order to sort $A[1..n]$, we recursively sort $A[1..n-1]$ and then insert $A[n]$ into the sorted array $A[1..n-1]$. Write a recurrence equation for the running time of this recursive version of insertion sort.